WHAT IS CLAIMED IS:

1. A method comprising:
providing a debugger suitable for a unit under test, wherein the unit under test includes a program under test;
causing the debugger to be loaded into the unit under test;
sending a plurality of test commands to the unit under test according to a test script; and
activating the debugger when a watched event occurs during execution of the program under test.

2. The method of claim 1 further comprising:
directing a debugger command to the debugger; and
recording information provided by the debugger according to the debugger command.

3. The method of claim 1 further comprising:
pausing execution of the program under test: and
allowing a user to control the debugger.

4. The method of claim 1 further comprising:
invoking the debugger while specifying the program under test as a target of the debugger.

5. The method of claim 1 further comprising:
instructing the debugger to associate itself with a process executing on the unit under test, wherein the process corresponds to the program under test.

6. The method of claim 1 further comprising:
sending a command to the debugger, wherein the command performs at least one of :
setting a breakpoint in the program under test;
setting a watchpoint in the program under test;
setting a catchpoint in the program under test;
setting a tracepoint in the program under test;

7. The method of claim 1 wherein the watched event includes at least one of a processor exception, a program under test error, reaching a breakpoint in the program under test; reaching a watchpoint in the program under test; reaching a catchpoint in the program under test; and reaching a tracepoint in the program under test.

8. The method of claim 1 further comprising:
selecting a platform-specific debugger corresponding to a processor in the unit under test; and
loading the platform-specific debugger into the unit under test.

9. The method of claim 8 further comprising:
loading, into the unit under test, a symbol file corresponding to the program under test.

10. A system comprising:
a memory;
a processor coupled to the memory; and
a debugger agent, wherein at least a portion of the debugger agent is encoded as instructions stored in the memory and executable on the processor, and wherein the debugger agent is configured to:
select a debugger program suitable for a unit under test, wherein the unit under test includes a program under test;
cause the debugger program to be loaded into the unit under test;
send a plurality of test commands to the unit under test according to a test script; and
activate the debugger program when a watched event occurs during execution of the program under test.

11. The system of claim 10 further comprising at least one debugger program stored in at least one of the memory and a storage device accessible by the processor.

12. The system of claim 10 further comprising at least one symbol file stored in at least one of the memory and a storage device accessible by the processor.

13. The system of claim 10 further comprising:

a test script handler, wherein at least a portion of the test script handler is encoded as instructions stored in the memory and executable on the processor.

14. The system of claim 13 wherein the test script handler is further configured to send the plurality of test commands to the debugger agent.

15. The system of claim 10 further comprising:

a second memory;

a second processor coupled to the second memory; and

a test script handler, wherein at least a portion of the test script handler is encoded as instructions stored in the second memory and executable on the second processor.

16. The system of claim 15 wherein the test script handler is further configured to send the plurality of test commands to the debugger agent.

17. The system of claim 10 wherein the debugger agent is further configured to:

direct a debugger program command to the debugger program; and

record information provided by the debugger program according to the debugger command.

18. The system of claim 10 wherein the debugger agent is further configured to:

suspend execution of the program under test: and

allow a user to control the debugger program.

19. The system of claim 10 wherein the debugger agent is further configured to:

invoke the debugger program while specifying the program under test as a target of the debugger program.

20. The system of claim 10 wherein the debugger agent is further configured to:

command the debugger program to associate itself with a process executing on the unit under test, wherein the process corresponds to the program under test.

21. The system of claim 10 wherein the debugger agent is further configured to:

send a command to the debugger program, wherein the command performs at least one of :

setting a breakpoint in the program under test;

setting a watchpoint in the program under test;

setting a catchpoint in the program under test;

setting a tracepoint in the program under test;

22. The system of claim 10 wherein the watched event includes at least one of a processor exception, a program under test error, reaching a breakpoint in the program under test; reaching a watchpoint in the program under test; reaching a catchpoint in the program under test; and reaching a tracepoint in the program under test.

23. The system of claim 10 wherein the debugger agent is further configured to:

select a platform-specific debugger program corresponding to a processor in the unit under test; and

load the platform-specific debugger program into the unit under test.

24. The system of claim 23 wherein the debugger agent is further configured to:

load, into the unit under test, a symbol file corresponding to the program under test.

25. A computer readable medium comprising program instructions executable on a processor, the computer readable medium being at least one of an electronic storage medium, a magnetic storage medium, an optical storage medium, and a

communications medium conveying signals encoding the instructions, wherein the program instructions are operable to implement each of:

providing a debugger suitable for a unit under test, wherein the unit under test includes a program under test;

causing the debugger to be loaded into the unit under test;

sending a plurality of test commands to the unit under test according to a test script; and

activating the debugger when a watched event occurs during execution of the program under test.

26. The computer readable medium of claim 25 further comprising program instructions operable to implement each of:

directing a debugger command to the debugger; and

recording information provided by the debugger according to the debugger command.

27. The computer readable medium of claim 25 further comprising program instructions operable to implement each of:

pausing execution of the program under test: and

allowing a user to control the debugger.

28. The computer readable medium of claim 25 further comprising program instructions operable to implement:

invoking the debugger while specifying the program under test as a target of the debugger.

29. The computer readable medium of claim 25 further comprising program instructions operable to implement:

instructing the debugger to associate itself with a process executing on the unit under test, wherein the process corresponds to the program under test.

30. The computer readable medium of claim 25 further comprising program instructions operable to implement:

sending a command to the debugger, wherein the command performs at least one of :

setting a breakpoint in the program under test;
setting a watchpoint in the program under test;
setting a catchpoint in the program under test;
setting a tracepoint in the program under test;

31. The computer readable medium of claim 25 wherein the watched event includes at least one of a processor exception, a program under test error, reaching a breakpoint in the program under test; reaching a watchpoint in the program under test; reaching a catchpoint in the program under test; and reaching a tracepoint in the program under test.

32. The computer readable medium of claim 25 further comprising program instructions operable to implement each of

selecting a platform-specific debugger corresponding to a processor in the unit under test; and

loading the platform-specific debugger into the unit under test.

33. The computer readable medium of claim 25 further comprising program instructions operable to implement:

loading, into the unit under test, a symbol file corresponding to the program under test.

34. An apparatus comprising:

a means for causing a means for debugging a program under test to be loaded into a unit under test, wherein the unit under test includes the program under test;

a means for sending a plurality of test commands to the unit under test according to a test script; and

a means for activating the means for debugging when a watched event occurs during execution of the program under test.

35. The apparatus of claim 34 further comprising:

a means for directing an instruction to the means for debugging a program under test; and

a means for recording information provided by the means for debugging a program under test.

36. The apparatus of claim 34 further comprising:
a means for pausing execution of the program under test: and
a means for allowing a user to control the means for debugging a program under test.

37. The apparatus of claim 34 further comprising:
a means for instructing the means for debugging a program under test to associate itself with a process executing on the unit under test, wherein the process corresponds to the program under test.

38. The apparatus of claim 34 further comprising:
a means for sending a command to the means for debugging a program under test , wherein the command performs at least one of :
setting a breakpoint in the program under test;
setting a watchpoint in the program under test;
setting a catchpoint in the program under test;
setting a tracepoint in the program under test;

39. The apparatus of claim 34 wherein the watched event includes at least one of a processor exception, a program under test error, reaching a breakpoint in the program under test; reaching a watchpoint in the program under test; reaching a catchpoint in the program under test; and reaching a tracepoint in the program under test.